

Számolás kettes számrendszerben

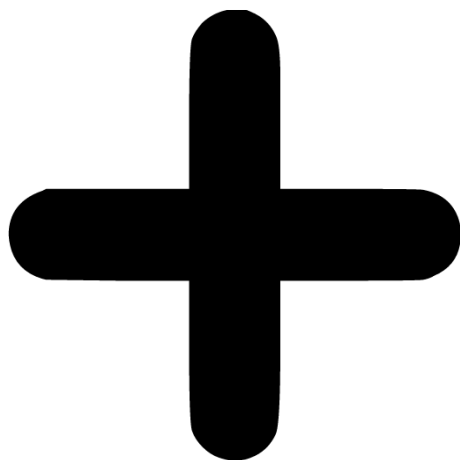
Mintapéldák összeadásra:

1. feladat: Végezze el a kijelölt műveletet a következő bináris számokkal:
 $0011\ 0111b + 0001\ 1110b$!

megoldás:

$$\begin{array}{r} 0011\ 0111\ b \quad 55\ d \\ +0001\ 1110\ b \quad 30\ d \\ \hline 0101\ 0101\ b \quad 85\ d \end{array}$$

$1+0 = 1,$	átvitel 0
$1+1 = 10,$	átvitel 1
$1+1+1 = 11$	átvitel 1
$1+0+1 = 10$	átvitel 1
$1+1+1 = 11$	átvitel 1
$1+0+1 = 10$	átvitel 1
$0+0+1 = 1$	átvitel 0
$0+0+0 = 0$	átvitel 0



összeadás egy bites számoknál

$$0 + 0 = 0$$

$$0 + 1 = 1$$

$$1 + 0 = 1$$

$$1 + 1 = 10 \text{ (átvitel)}$$

Számolás kettes számrendszerben

Mintapéldák kivonásra:

A módszer ugyanaz lesz, amit a tízes számrendszerben megszokhattunk:

4 feladat: Végezze el a kijelölt műveletet 0011 0111b - 0001 1110b bináris számokkal!

megoldás:

0011 0111 b	55 d
- 0001 1110 b	30 d
<hr/>	
0001 1001 b	15 d

1-0 = 1, átvitel 0
1-1 = 0, átvitel 0
1-1 = 0 átvitel 0
0-1 = 1 átvitel 1 (a kivonás valójában így néz ki: 10-1=1, és az átvitel ezért keletkezik)
1-1-1 = 1 átvitel 1
1-1= 0 átvitel 0
0-0 = 0, átvitel 0
0-0 = 0, átvitel 0

kivonás

$$0 - 0 = 0$$

$$0 - 1 = -1 \text{ (a különbség 1, átvitel 1)}$$

$$1 - 0 = 1$$

$$1 - 1 = 0$$

$$0 - 1 \text{ 1 átvittel} = 0 \text{ az átvitel 1}$$

$$1 - 1 \text{ 1 átvittel} = 1 \text{ az átvitel 1}$$

Kivonás komplementesképzéssel

A számítógép tervezésekor arra törekedtek, hogy minél kevesebb műveletet kelljen ismernie a számítógép központi egységének.

Az összes műveletet az összeadásra vezették vissza.

Az összeadással elvégzett kivonás előtt a negatív számokat speciális formára, komplementessé kell alakítani.

Példaként nézzük meg a $-168_{(10)}$ -as, azaz a $10101000_{(2)}$ szám átalakítását.

<i>Lépés</i>	<i>Művelet</i>	<i>Eredmény</i>
1.	abszolút érték felírása	$10101000_{(2)}$
2.	egyes komplement képzése	$01010111_{(2)}$

A biteket ellenkezőjére váltjuk.

Az egyes komplement képzésekor problémaként vetődik fel, hogy így a nullára két kód is van:

$$00000000_{(2)} = "+0",$$

$$11111111_{(2)} = "-0"$$

Erre megoldást kínál a kettes komplement képzés.

Kettes komplementens képzése

Lépés	Művelet	Eredmény
1.	abszolút érték felírása	10101000 ₍₂₎
2.	egyes komplementens képzése	01010111 ₍₂₎
3.	kettes komplementens képzése	01011000 ₍₂₎

Az egyes komplementenshez 1-et hozzáadunk.

Ezek után nézzünk konkrét példát a bináris kivonásra. Végezzük el a következő műveletet: $184_{(10)} - 100_{(10)}$

Lépés	Művelet	Eredmény
1.	$-100_{(10)}$ abszolút értékének felírása	01100100 ₍₂₎
2.	A kettes komplementens meghatározása	10011100 ₍₂₎
3.	A kivonás elvégzése:	
	$\begin{array}{r} 184 \\ -100 \\ \hline 84_{(10)} \end{array}$	$\begin{array}{r} 10111000 \\ + 10011100 \\ \hline 101010100_{(2)} \end{array}$

Túlcsordult számjegy.

Elhagyjuk a túlcsordult jegyet, ha van, így az eredmény: **01010100**

Számolás kettes számrendszerben


Mintapéldák szorzásra:

Az osztást és a szorzást csak bináris számokkal tárgyaljuk!

Látható lesz, hogy a kettes számrendszerben elvégzett szorzás a legegyszerűbb műveletek egyike, hiszen vagy egyel vagy nullával kell szoroznunk.

7. feladat: Végezze el a kijelölt műveletet $0011\ 0111b * 0000\ 1010b$ bináris számokkal! A műveletvégzésnél a számok elején található vezető nullákat elhagytuk, hiszen a műveletvégzést nem befolyásolják, csak a részművelet végrehajtási számot növelik.

megoldás:


$$\begin{array}{r} 11\ 01\ 11 * \quad 1010 \\ 11\ 01\ 11 \\ 0\ 00\ 00\ 0 \\ \quad 11\ 01\ 11 \\ + \quad 0\ 00\ 000 \\ \hline 1\ 00\ 01\ 00\ 110 \end{array}$$

$$\begin{array}{r} 55\ d * 10\ d \\ 55 \\ + 00 \\ \hline 550\ d \end{array}$$

Számolás kettes számrendszerben

Osztás:

A jobbra történő eltolás a kettes számrendszerben a kettő negatív hatványával való szorzás, de ha jobban belegondolunk, ez nem jelent mást, mint osztás egy (1 0) alakú számmal.

Osztásnál alapvetően két esetet különböztetünk meg, a maradék nélküli és a maradékos osztást, lássunk most mindkettőre példát:

8. feladat: Végezze el a kijelölt műveletet $0011\ 0110b : 0000\ 0010b$

megoldás:

$$0011\ 0110b : 0000\ 0010b = 0001\ 1011b \\ \text{maradék: } 0$$

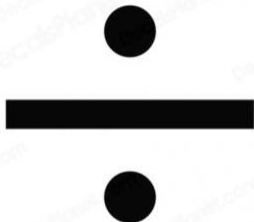
ell:

$$54\ d : 2d = 27d$$

9. feladat: Végezze el a kijelölt műveletet $0011\ 0111b : 0000\ 0100b !$

megoldás:

$$0011\ 0111b : 0000\ 0100b = 0000\ 1101b \\ \text{maradék } 11b\ (3d)$$



Tört számok binárisan

- Kettes számrendszerben ábrázolt törtek értéke:

$$0,1011_{(2)} = ?$$

A "kettedes" vessző/pont utáni helyiértékek 2-nek a negatív hatványai. Vagyis:

$$0,1011_{(2)} = 2^{-1} + 2^{-3} + 2^{-4} = 0,5 + 0,125 + 0,0625 = 0,6875$$

- Tizedes tört átváltása bináris törtté (kettővel osztások helyett kettővel szorzásokkal!)

- A számot (tehát az eredeti szám törtrészét!) szorozzuk 2-vel!
- A keletkezett szám egész részét (úgy ahogy az egészek átváltásakor a maradékokat) írjuk ki!
- A keletkezett szám **törtrészével(!!!)** folytassuk a műveletet, míg a törtrész 0 nem lesz, vagy míg el nem érjük a kellő számú "kettedes"-jegyet!
- Az egész részeket **fentről lefelé** kell a "kettedes" vessző után írni.

Tört számok binárisan

- Írjuk fel a 0,6875 számot binárisan!

		0,6875
	$0,6875 * 2 = 1,375$	Ennek az egész része 1.
1,375	$0,375 * 2 = 0,75$	Ennek az egész része 0.
0,75	$0,75 * 2 = 1,5$	Ennek az egész része 1.
1,5	$0,5 * 2 = 1,0$	Ennek az egész része 1.
1,0	$0 * 2 = 0$	Itt vége az osztásnak.
Tehát: $0,6875 = 0,1011_{(2)}$		

Hexadecimális összeadástábla

+	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
1	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	10
2	2	3	4	5	6	7	8	9	A	B	C	D	E	F	10	11
3	3	4	5	6	7	8	9	A	B	C	D	E	F	10	11	12
4	4	5	6	7	8	9	A	B	C	D	E	F	10	11	12	13
5	5	6	7	8	9	A	B	C	D	E	F	10	11	12	13	14
6	6	7	8	9	A	B	C	D	E	F	10	11	12	13	14	15
7	7	8	9	A	B	C	D	E	F	10	11	12	13	14	15	16
8	8	9	A	B	C	D	E	F	10	11	12	13	14	15	16	17
9	9	A	B	C	D	E	F	10	11	12	13	14	15	16	17	18
A	A	B	C	D	E	F	10	11	12	13	14	15	16	17	18	19
B	B	C	D	E	F	10	11	12	13	14	15	16	17	18	19	1A
C	C	D	E	F	10	11	12	13	14	15	16	17	18	19	1A	1B
D	D	E	F	10	11	12	13	14	15	16	17	18	19	1A	1B	1C
E	E	F	10	11	12	13	14	15	16	17	18	19	1A	1B	1C	1D
F	F	10	11	12	13	14	15	16	17	18	19	1A	1B	1C	1D	1E

Hexadecimális számok összeadása

- Számoljuk ki: $(B A 3)_{16} + (5 D E)_{16}$
- $3+E=11$ leírjuk az **1**-et, átvitel (marad) 1
- $A+D=17$ és volt 1 átvitel, az összesen 18, leírjuk a **8**-at és maradt 1 átvitel
- $B+5=10$ és volt 1 átvitel, az összesen 11, leírjuk a **11**-et.
- Tehát az eredmény **1181**.

Hexadecimális szorzótábla

Table for Hexadecimal Multiplication

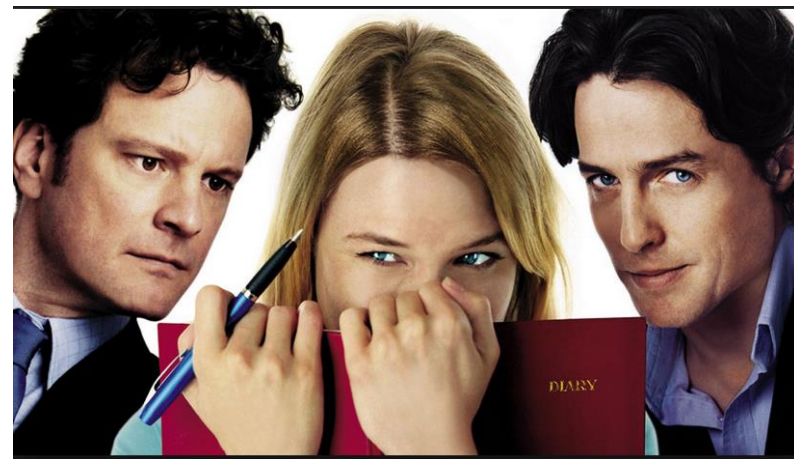
X	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
2	0	2	4	6	8	A	C	E	10	12	14	16	18	1A	1C	1E
3	0	3	6	9	C	F	12	15	18	1B	1E	21	24	27	2A	2D
4	0	4	8	C	10	14	18	1C	20	24	28	2C	30	34	38	3C
5	0	5	A	F	14	19	1E	23	28	2D	32	37	3C	41	46	4B
6	0	6	C	12	18	1E	24	2A	30	36	3C	42	48	4E	54	5A
7	0	7	E	15	1C	23	2A	31	38	3F	46	4D	54	5B	62	69
8	0	8	10	18	20	28	30	38	40	48	50	58	60	68	70	78
9	0	9	12	1B	24	2D	36	3F	48	51	5A	63	6C	75	7E	87
A	0	A	14	1E	28	32	3C	46	50	5A	64	6E	78	82	8C	96
B	0	B	16	21	2C	37	42	4D	58	63	6E	79	84	8F	9A	A5
C	0	C	18	24	30	3C	48	54	60	6C	78	84	90	9C	A8	B4
D	0	D	1A	27	34	41	4E	5B	68	75	82	8F	9C	A9	B6	C3
E	0	E	1C	2A	38	46	54	62	70	7E	8C	9A	A8	B6	C4	D2
F	0	F	1E	2D	3C	4B	5A	69	78	87	96	A5	B4	C3	D2	E1

Hexadecimális számok szorzása

- Számoljuk ki: $1A8 * AF$
- $F*8 = 78$, leírjuk a **8**-at, átvitel 7
- $F*A=96$, és maradt 7, az összesen (hexaban összeadva) 9D, leírjuk a **D**-t és marad 9.
- $F*1=F$ és maradt 9, az összesen **18**
- **Az F-vel szorzás eredménye: 18D7**
- $A*8=50$, leírjuk a **0**-t, maradt 5
- $A*A=64$, maradt 5, az összesen (hexaban összeadva) 69, leírjuk a **9**-t, maradt 6
- $A*1=A$ és maradt 6, az összesen (hexaban összeadva) **10**
- **Az A-val szorzás eredménye: 1090**
- **18D7**
+1090
- = **121D7** (hexaban kellett összeadni)

Kódolási feladatok

- Írjuk le a teljes nevünket ASCII kódban!
- Írjuk le a kedvenc gyümölcsünk nevét bináris kódban!
- Írjuk le az utca nevét, ahol lakunk decimális kódban!
- Írjuk le a kedvenc filmünk nevét hexadecimális kódban!
- Adjuk oda a szomszédunknak, aki próbálja megfejteni!



ASCII kódtáblázat

0		30	▲	60	<	90	Z	120	x	150	ı	180	ı	210	Ď	240	
1	☺	31	▼	61	=	91	[121	y	151	Š	181	Á	211	Ě	241	"
2	☻	32		62	>	92	\	122	z	152	š	182	Â	212	ď	242	´
3	♥	33	!	63	?	93]	123	{	153	Ö	183	Ë	213	Ň	243	˘
4	♦	34	"	64	@	94	^	124		154	Ü	184	Ş	214	í	244	˙
5	♣	35	#	65	A	95	_	125	}	155	Ť	185	ı	215	î	245	§
6	♠	36	\$	66	B	96	`	126	~	156	ř	186		216	ě	246	+
7	•	37	%	67	C	97	a	127	△	157	ł	187	ı	217	ǰ	247	˚
8	◻	38	&	68	D	98	b	128	Ç	158	×	188	ı	218	ǂ	248	◦
9	○	39		69	E	99	c	129	ü	159	č	189	Ž	219	■	249	˝
10	◼	40	(70	F	100	d	130	é	160	á	190	ž	220	■	250	ˆ
11	♂	41)	71	G	101	e	131	â	161	í	191	ı	221	ı	251	ű
12	♀	42	*	72	H	102	f	132	ä	162	ó	192	ı	222	Ü	252	Ř
13	♪	43	+	73	I	103	g	133	û	163	ú	193	ı	223	■	253	ř
14	♫	44	,	74	J	104	h	134	ć	164	Ą	194	ı	224	Ó	254	■
15	☀	45	-	75	K	105	i	135	ç	165	ą	195	ı	225	β	255	
16	▶	46	.	76	L	106	j	136	ı	166	Ż	196	ı	226	Ö		
17	◀	47	alt	77	M	107	k	137	ë	167	ż	197	ı	227	Ń		
18	↑	48	0	78	N	108	l	138	Ï	168	Ę	198	ı	228	ń		
19		49	1	79	O	109	m	139	ö	169	ę	199	ı	229	ñ		
20	ı	50	2	80	P	110	n	140	î	170	ı	200	ı	230	Š		
21	§	51	3	81	Q	111	o	141	Ž	171	ż	201	ı	231	š		
22	—	52	4	82	R	112	p	142	Ä	172	Č	202	ı	232	Ř		
23	↑	53	5	83	S	113	q	143	Ć	173	ş	203	ı	233	Ú		
24	↑	54	6	84	T	114	r	144	É	174	«	204	ı	234	ı		
25	↓	55	7	85	U	115	s	145	Ĺ	175	»	205	ı	235	Ů		
26	→	56	8	86	V	116	t	146	Í	176	ı	206	ı	236	ý		
27	←	57	9	87	W	117	u	147	ô	177	ı	207	ı	237	Ý		
28	L	58	:	88	X	118	v	148	ö	178	ı	208	ı	238	ı		
29	↔	59	;	89	Y	119	w	149	Ł	179	ı	209	ı	239	ı		

Feladatok összeadásra

Adja össze a következő bináris számokat:

1	101011	+	1111
2	101000	+	101
3	1111001	+	1011
4	101010	+	10101
5	1111	+	01

Adja össze a következő oktális számokat:

6	777	+	1
7	2356	+	44
8	100	+	44
9	167	+	167
10	111	+	722

Adja össze a következő hexadecimális számokat:

11	ABCD	+	1111
12	100	+	1B
13	15B2	+	C4F
14	333	+	DDD
15	1A2B3C	+	68F

Megoldások összeadásra

Megoldások:

1	101011	+	1111	=	111010
2	101000	+	101	=	101101
3	1111001	+	1011	=	10000100
4	101010	+	10101	=	111111
5	1111	+	01	=	10000
6	777	+	1	=	1000
7	2356	+	44	=	2422
8	100	+	44	=	144
9	167	+	167	=	356
10	111	+	722	=	1033
11	ABCD	+	1111	=	BCDE
12	100	+	1B	=	11B
13	15B2	+	C4F	=	2201
14	333	+	DDD	=	1110
15	1A2B3C	+	68F	=	1A31CB

Feladatok kivonásra

Határozza meg a következő bináris számok különbségét:

1	11111	-	1111
2	110011	-	1010
3	101010	-	10101
4	100000	-	1
5	111000	-	1011

Határozza meg a következő oktális számok különbségét:

6	1000	-	100
7	1574	-	456
8	123	-	55
9	564	-	12
10	455	-	366

Határozza meg a következő hexadecimális számok különbségét:

11	ABC	-	999
12	FFDD	-	AAAA
13	4E5D	-	147
14	EE9	-	94
15	100	-	1

Megoldások kivonásra

Megoldások:

1	11111	-	1111	=	10000
2	110011	-	1010	=	101101
3	101010	-	10101	=	10101
4	100000	-	1	=	11111
5	111000	-	1011	=	101101
6	1000	-	100	=	700
7	1574	-	456	=	1116
8	123	-	55	=	46
9	564	-	12	=	552
10	455	-	366	=	67
11	ABC	-	999	=	123
12	FFDD	-	AAAA	=	5544
13	4E5D	-	147	=	4D19
14	EE9	-	94	=	E55
15	100	-	1	=	FF

Feladatok szorzásra, osztásra

Végezze el a kijelölt szorzásokat:

1	11011	*	10
2	11111	*	10000
3	1001101	*	101
4	10110	*	110
5	1111	*	11

Végezze el a kijelölt osztásokat:

6	11001000	:	100
7	11110	:	10
8	1101001	:	101
9	1010111	:	111
10	1010111	:	1

Megoldások szorzásra, osztásra

Megoldások:

1	11011	*	10	=	110110
2	11111	*	10000	=	111110000
3	1001101	*	101	=	110000001
4	10110	*	110	=	10000100
5	1111	*	11	=	101101
6	11001000	:	100	=	110010
7	11110	:	10	=	1111
8	1101001	:	101	=	10101
9	1010111	:	111	=	1100 maradék 11
10	1010111	:	1	=	1010111

Feladatok

Végezze el az alábbi átalakításokat!

1. $2010_{10} = ?_{\underline{8}}$
2. $2010_{10} = ?_{\underline{2}}$
3. $2010_{10} = ?_{\underline{16}}$
4. $111010011_{\underline{2}} = ?_{\underline{10}}$
5. $2010_{\underline{8}} = ?_{\underline{10}}$
6. $2010_{\underline{16}} = ?_{\underline{10}}$
7. $111010011_{\underline{2}} = ?_{\underline{8}}$
8. $2010_{\underline{8}} = ?_{\underline{2}}$
9. $2010_{\underline{16}} = ?_{\underline{8}}$

Végezze el az alábbi átalakításokat!

1. $1011010011_{\underline{2}} = ?_{\underline{10}}$
2. $1011010011_{\underline{2}} = ?_{\underline{8}}$
3. $1011010011_{\underline{2}} = ?_{\underline{16}}$
4. $E3A_{\underline{16}} = ?_{\underline{10}}$
5. $E3A_{\underline{16}} = ?_{\underline{8}}$
6. $E3A_{\underline{16}} = ?_{\underline{2}}$
7. $732_{\underline{8}} = ?_{\underline{10}}$
8. $732_{\underline{8}} = ?_{\underline{16}}$
9. $732_{\underline{8}} = ?_{\underline{2}}$

A számok ábrázolása a számítógépben

a) Fixpontos ábrázolás

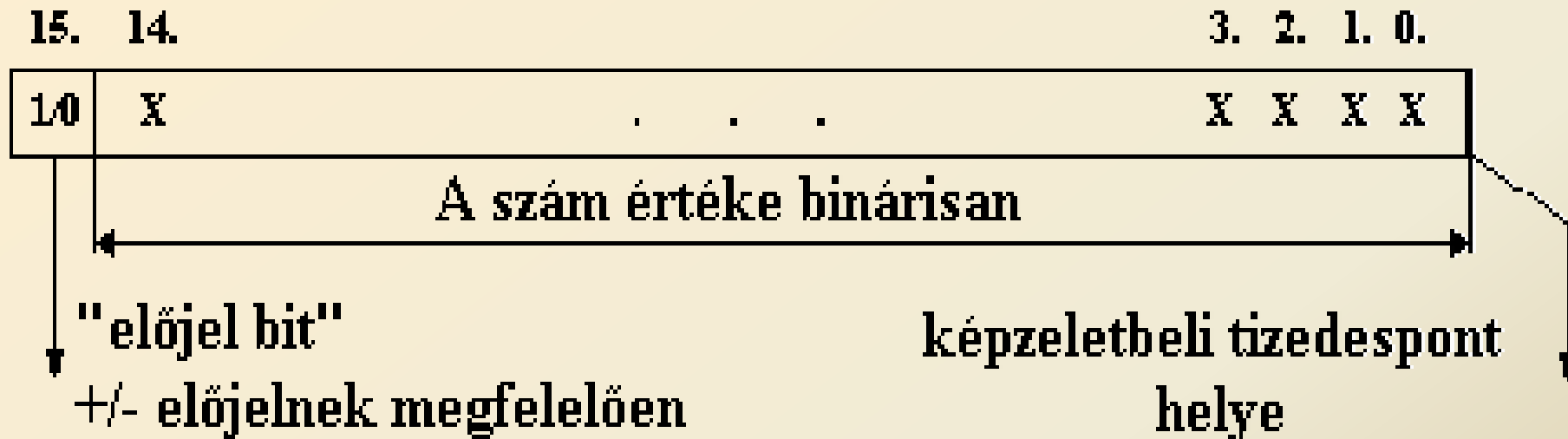
Ez a számok műveletvégzésre alkalmas formában történő tárolására szolgál.

Pozitív és negatív egész számok ábrázolására. A negatív számokat a már ismertetett kettes komplementum szerint értelmezik a gépek.

Törtrésszel rendelkező számokat is ábrázolhatunk, de ekkor a törtet jelző pont csak logikailag létezik, a számítógép nem "teszi ki", helyét nem változtatja. Nyomtatásnál az elhelyezéséről a programozónak kell gondoskodnia.

A számok ábrázolása a számítógépben

a) Fixpontos ábrázolás



A fixpontos számábrázolás hátrányai:

- az ábrázolható tartomány kicsi: 2 Byte-on a legnagyobb 32.767, a legkisebb -32.768
- a számok pontossága erősen korlátozott:
ha egész számot ábrázol $7/4 = 1$ és a $4/4 = 1$

A számok ábrázolása a számítógépben

b) Lebegőpontos számábrázolás

A fixpontos hátrányait kiküszöbölő, a számok hatványkitevős (matematikában használt normál alakhoz hasonló) felírásán alapuló számábrázolás.

Például:

$$175_{(10)} = 0.175 * 10^3 \quad 10110011_{(2)} = 0.10110011 * 2^8$$

$$0.375_{(10)} = 0.375 * 10^0 \quad 0.011_{(2)} = 0.11 * 2^{-1}$$

A számok ábrázolása a számítógépben

b) Lebegőpontos számábrázolás

Általánosan felírva: $A = M * p^k$

A = az eredeti szám

M = az együttható,

ennek a tört része az ún. mantissza

p = a hatvány alapja

k = a hatvány kitevője, az ún. karaktisztika

Ebből észrevehetjük, hogy néhány elem minden szám esetén ismétlődik, ezért ezeket a számítógépen nem kell külön ábrázolni.

A számok ábrázolása a számítógépben

b) Lebegőpontos számábrázolás

Mi hagyható el?

- 0 és a pont (.)
- a hatvány alapja

Mi az ami megmarad?

- mantissza ill. a mantissza előjele
- karakterisztika ill. a karakterisztika előjele

Például: $175_{(10)} = 0.175 * 10^3$ $10110011_{(2)} = 0.10110011 * 2^8$
 $0.375_{(10)} = 0.375 * 10^0$ $0.011_{(2)} = 0.11 * 2^{-1}$

A számok ábrázolása a számítógépben

b.) Lebegőpontos számábrázolás

A számokat 6 bájtos valós típusú mennyiségként ábrázoljuk

A mantissa egészrésze

Pl.: -23.1875

23		1	(23÷2=	11	marad:	1)
11		1	(11÷2=	5	marad:	1)
5		1	(5÷2=	2	marad:	1)
2		0	(2÷2=	1	marad:	0)
1		1	(1÷2=	0	marad:	1)
0							

A mantissa törtrésze

0,1875		0	(0,1875×2=	0,375)
0,375		0	(0,375 ×2=	0,75)
0,75		1	(0,75 ×2=	1,5)
0,5		1	(0,5 ×2=	1,0)
0					

$$X=m*2^k \text{ ahol } 0,5 \leq |m| \leq 1$$

Mantissza: m valós szám

Karakterisztika: k egész szám

A számok ábrázolása a számítógépben

b.) Lebegőpontos számábrázolás

A számokat 6 bájtos valós típusú mennyiségként ábrázoljuk

Pl.: -23.1875

$$X = m \cdot 2^k \text{ ahol } 0,5 \leq |m| \leq 1$$


A mantissza egészrésze = **10111**

Mantissza: m valós szám


A mantissza törtrésze = 0011

Karakterisztika: k egész szám

$$|m| = 10111.0011$$

10111.001 | 1000000 | 00000000 | 00000000 | 00000000 | kkkkkkkk


$$|m| = .101110011$$

.10111001 | 1000000 | 00000000 | 00000000 | 00000000 | kkkkkkkk


$$|m| = 0.101110011$$

$k = 5$ Normálás 0.1xxxx A pontot 5 lépéssel balra tettük

A számok ábrázolása a számítógépben

b.) Lebegőpontos számábrázolás

A mantissa előjele

$$|m| = .101110011$$

$$.10111001 | 1000000 | 00000000 | 00000000 | 00000000 | kkkkkkkk$$

Ha a mantissa
negatív, akkor legyen 1
különben legyen 0

Mivel itt mindig 1 állna,
ezért ezt a bitet ruházzuk
fel a mantissa előjelének
jelentésével

$$X = m \cdot 2^k \text{ ahol } 0,5 \leq |m| \leq 1$$

Mantissza: m valós szám

Karakterisztika: k egész szám

$$|m| = 0.101110011$$

$$k = 5$$

A számok ábrázolása a számítógépben

b.) *Lebegőpontos számábrázolás* A karakterisztika (kitevő) előjele

A karakterisztikát az ábrázolt legkisebb szám abszolút értékével megnöveljük, azaz hozzá adunk $|-128|=128$ -at, majd átváltjuk 2-es számrendszerbe.

Tehát ha $k=5$,
akkor $k:=5+128=133$

$$X=m*2^k \text{ ahol } 0,5 \leq |m| \leq 1$$

Mantissza: m valós szám

Karakterisztika: k egész szám

Ezt az alakot nevezzük 128-cal eltolt nullpontú ábrázolásnak

A számok ábrázolása a számítógépben

b.) Lebegőpontos számábrázolás

A karakterisztika (kitevő) előjele

$$X = m \cdot 2^k \text{ ahol } 0,5 \leq |m| \leq 1$$

A karakterisztika

133	1	(133 ÷ 2 =	66	marad: 1)
66	0	(66 ÷ 2 =	33	marad: 0)
33	1	(33 ÷ 2 =	16	marad: 1)
16	0	(16 ÷ 2 =	8	marad: 0)
8	0	(8 ÷ 2 =	4	marad: 0)
4	0	(4 ÷ 2 =	2	marad: 0)
2	0	(2 ÷ 2 =	1	marad: 0)
1	1	(1 ÷ 2 =	0	marad: 1)
0				

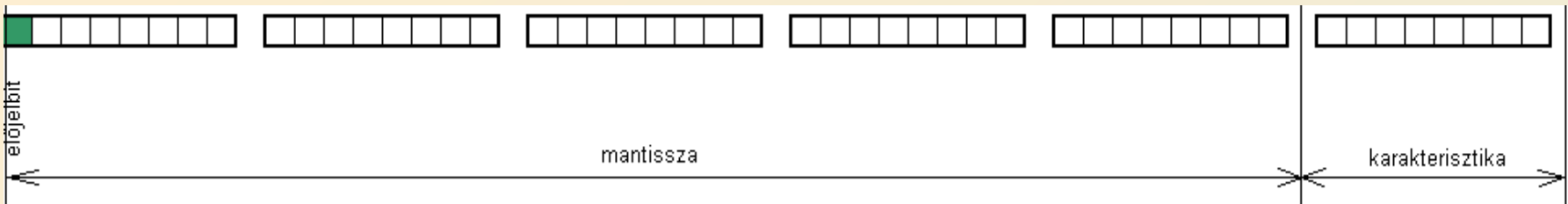
Mantissza: m valós szám

Karakterisztika: k egész szám

k=10000101

A számok ábrázolása a számítógépben

A 6 bájtos normált ábrázolás eddigiek ismeretében:



fx	A	B	C	D	E	F
1	-23.1875	lebegőpontos ábrázolása 6 bájton				
2	10111001	10000000	00000000	00000000	00000000	10000101
3	balról jobbra kitöltjük a kapott bináris szám jegyeivel és a maradék helyekre 0-t teszünk					k + 128 binárisan
4	Első jegyet nem változtatjuk, mert negatív szám volt, különben 0-ra kellene a legvégén átállítani.					

Ábrázolható tartomány:

konvenció (megállapodás): $0 := \{k = -128, \text{mantissza} = \text{tetszőleges}\}$

A legkisebb pozitív valós szám: $0.5 * 2^{-127} \approx 2,938 * 10^{-39}$

Példa **negatív** szám ábrázolására (első bit 1 marad)

	A	B	C	D	E	F	G	H	I	J	K	L	M	N
1	Adjuk meg 6 bájt az alábbi számokat													
2														
3	1.		-6,625	- 6 kettes számrendszerben:				6	2		3	6	0	
4			negatív szám			-110		3	2		1	2	1	
5								1	2		0	0	1	
6								0	2		0	0 ide nem írunk semmit		
7					0,625 kettes számrendszerben									
8						0,101			0,625		2	1,250	1	
9									0,250		2	0,500	0	
10					- 6,625 kettes számrendszerben					0,500		2	1,000	1
11						-110,101			0,000		2	0,000	0	
12														
13														
14					mantissza	alap		karakterisztika					lebegőpontos alak	
15					-0,110101		2	3	mert		-110,101	=	-0,110101 * 2^3	
16														
17					tehát k= 3									
18					utolsó bájt: +3+128 = 131 binárisan				131		2	65	130	1
19									65		2	32	64	1
20									32		2	16	32	0
21									16		2	8	16	0
22									8		2	4	8	0
23									4		2	2	4	0
24									2		2	1	2	0
25									1		2	0	0	1
26									0		2	0	0 ide nem írunk semmit	
27					utolsó bájt	10000011								
28														
29					11010100	00000000	00000000	00000000	00000000		10000011			
30					negatív szám volt, ezért az első számjegyet nem változtatjuk meg:									
31					- 6,625 6 bájtos ábrázolása tehát									
32														
33					11010100	00000000	00000000	00000000	00000000		10000011			

Példa pozitív szám ábrázolására (első bit 0 lesz)

	A	B	C	D	E	F	G	H	I	J	K	L	M	N
36	2.		123,375		123 kettes számrendszerben			123	2	61	122	1		
37			pozitív szám			1111011		61	2	30	60	1		
38								30	2	15	30	0		
39								15	2	7	14	1		
40								7	2	3	6	1		
41								3	2	1	2	1		
42								1	2	0	0	1		
43								0	2	0	0	0	ide nem írunk semmit	
44														
45					0,375 kettes számrendszerben			0,37500	2	0,75	0			
46						0,011		0,75000	2	1,5	1			
47								0,50000	2	1	1			
48								0,00000	2	0	0			
49														
50					123,375 kettes számrendszerben									
51						1111011,011								
52														
53					mantissza	alap	karakterisztika						lebegőpontos alak	
54					0,1111011011	2	7	mert		1111011,011	=	0,1111011011 * 2^7		
55														
56					tehát k= 7									
57					utolsó bájt: +7+128 = 135 binárisan			135	2	67	134	1		
58								67	2	33	66	1		
59								33	2	16	32	1		
60								16	2	8	16	0		
61								8	2	4	8	0		
62								4	2	2	4	0		
63								2	2	1	2	0		
64								1	2	0	0	1		
65								0	2	0	0	ide nem írunk semmit		
66					utolsó bájt	10000111								
67														
68					11110110	110000000	00000000	00000000	00000000	00000000	10000111			
69					pozitív szám volt, ezért az első számjegyet 0-ra változtatjuk									
70					123,375 6 bájtos ábrázolása tehát									
71														
72					01110110	110000000	00000000	00000000	00000000	00000000	10000111			

ANSI C adattípusok

Tipus	Méret bájtban (minimum)	Határok:
Egész		
char	1	-128 - 127
unsigned char	1	0 - 255
int	2	-32768 - 32767
unsigned int	2	0 - 65535
long	4	-2.147.483.647 - 2.147.483.647
unsigned long	4	0 - 4.294.967.295
Lebegőpontos		
float	4	$\pm 3.4 \cdot 10^{-38}$ - $\pm 3.4 \cdot 10^{38}$ 6-7 decimális jegy pontosság
double	8	$\pm 1.7 \cdot 10^{-308}$ - $\pm 1.7 \cdot 10^{308}$ 15-16 decimális jegy pontosság
long double	10	$\pm 1.2 \cdot 10^{-4932}$ - $\pm 1.2 \cdot 10^{4932}$ 19 decimális jegypontosság

Adattípusok a Java programozási nyelvben

A Java nyelvben az adattípusoknak két csoportja van: **primitív** és **referencia** típusok. A primitív adattípusok egy egyszerű értéket képesek tárolni: számot, karaktert vagy logikai értéket.

Egészek

Típus	Leírás	Méret/formátum
byte	bájt méretű egész	8-bit kettes komplement
short	rövid egész	16-bit kettes komplement
int	egész	32-bit kettes komplement
long	hosszú egész	64-bit kettes komplement

Adattípusok a Java programozási nyelvben

Valós számok

Típus	Leírás	Méret/formátum
<u>float</u>	egyszeres pontosságú lebegőpontos	32-bit IEEE 754
<u>double</u>	dupla pontosságú lebegőpontos	64-bit IEEE 754

Egyéb típusok

Típus	Leírás	Méret/formátum
<u>char</u>	karakter	16-bit Unicode karakter
<u>boolean</u>	logikai érték true vagy false	

IEEE 754 standard

single 32 bites, double 64 bites (, extended 80 bites).

típus	előjel	kitevőrész	törtrész
single	1 bit	8 bit 127-többletes	23 bit
double	1 bit	11 bit 1023-többletes	52 bit

single: Ha $0 < a \text{ kitevőrész} < 255$, a szám normalizált.

Normalizált tört vezető 1-es bitje nincs ábrázolva!

Normalizált számok (IEEE 754, single)

$$0 < \text{kitevőrész} < 255$$

kitevőrész = kitevő + 127, 127 többletes.

Lehetséges kitevők: -126, -125, ... , +127.

Közvetlenül a törtrész elé kell képzelni egy *1*-est
(**implicit bit**) és a bináris pontot.

Az ábrázolt szám: $\pm (1 + \text{törtrész}) * 2^{\text{kitevő}}$

Pl.: 1 0011 1111 1000 ... 0000₂ = 3F80 0000₁₆

 0,5 0011 1111 0000 ... 0000₂ = 3F00 0000₁₆

 -1,5 1011 1111 1100 ... 0000₂ = BFC0 0000₁₆

 ± kitevőrész 1. törtrész

Normalizálatlan számok (IEEE 754, single)

Ha a kitevőrész = 0

Ilyenkor a kitevő -126 (! nem 127),
a bináris pontot implicit 0 előzi meg (nincs ábrázolva).

Az ábrázolt szám: $\pm (\text{tötrész}) * 2^{-126}$

$$\begin{aligned} \text{Pl.: } 2^{-127} &= 2^{-126} * 0,100 \dots 0000_2 = \\ &= \underbrace{0000 \ 0000 \ 0100 \ \dots \ 0000}_2 = 0040 \ 0000_{16} \\ &\quad \pm \text{ kitevőrész } 0. \text{ tötrész } (2^{-1}) \end{aligned}$$

$$\begin{aligned} - 2^{-149} &= - 2^{-126} * 0,000 \dots 0001_2 = \\ &= \underbrace{1000 \ 0000 \ 0000 \ \dots \ 0001}_2 = 8000 \ 0001_{16} \\ &\quad \pm \text{ kitevőrész } 0. \text{ tötrész } (2^{-23}) \end{aligned}$$

A legkisebb pozitív (single) normalizált szám:

$$\begin{aligned} 2^{-126} &= 2^{-126} * 1,000 \dots 0000_2 = \\ &= \underbrace{0000 \ 0000 \ 1000 \ \dots \ 0000}_2 = 0080 \ 0000_{16} \\ &\quad \pm \text{kitevőrész } 1. \text{ törtrész} \end{aligned}$$

A legnagyobb pozitív (single) normalizálatlan szám:

$$\begin{aligned} 2^{-126} * 0,111 \dots 1111_2 &= \\ &= \underbrace{0000 \ 0000 \ 0111 \ \dots \ 1111}_2 = 007F \ FFFF_{16} \\ &\quad \pm \text{kitevőrész } 0. \text{ törtrész} \\ &\approx 2^{-126} \end{aligned}$$

A különbségük csupán 2^{-149} .

Normalizálatlan számok (IEEE 754, single)

Ha a kitevőrész = 255

Túl nagy számok (túlcsordulás):

- ∞ (végtelen): pl. $1/0$,
- **NaN** (Not a Number): pl. ∞ / ∞

Normalizált	\pm	$0 < \text{kitevőrész} < \text{Max}$	bitminta
Nem normalizált	\pm	0	nem nulla bitminta
Nulla	$+$	0	0
Végtelen	\pm	111...1	0
Nem szám	\pm	111...1	nem nulla bitminta

(IEEE 754, single)

Logikai műveletek

A számítógép nem csak - a számrendszereknél ismertetett - matematikai műveletek, hanem logikai műveletek végrehajtására is képes.

A logikában állítások vannak, melyek vagy igazak vagy hamisak. Ennek megfelelően a logikai adatok két értéket vehetnek fel: ha igaz, az értéke 1, ha hamis, az értéke 0. A logikai adatok ábrázolása általában 1Byte-on történik:

- 00000001 = logikai igaz
- 00000000 = logikai nem

Ismerkedjünk meg néhány logikai művelettel:

Logikai műveletek

A: Esik az eső

B: Van nálam ernyő

C: Megázom

ÉS (konjunkció): A and B

Esik az eső és van nálam ernyő

VAGY (diszjunkció): B or C

Vagy van nálam ernyő vagy megázom

TAGADÁS (negáció): not B

nincs nálam ernyő

KÖVETKEZTETÉS (implikáció): (A and not B) then C

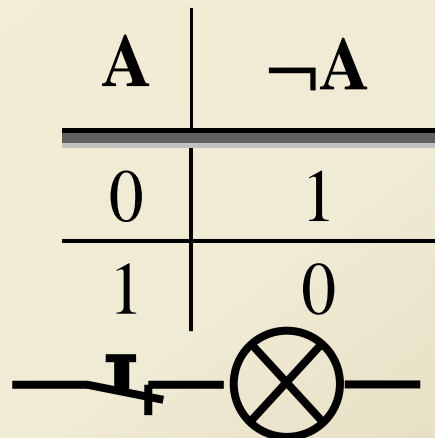
Ha esik az eső és nincs nálam ernyő, akkor megázom



Logikai műveletek

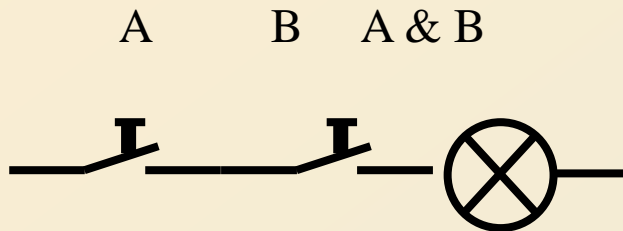
a) Negálás (NOT)

Olyan művelet, amely során az igazból hamisra, a hamisból igazra váltunk.



Logikai műveletek

b) ÉS kapcsolat (AND)

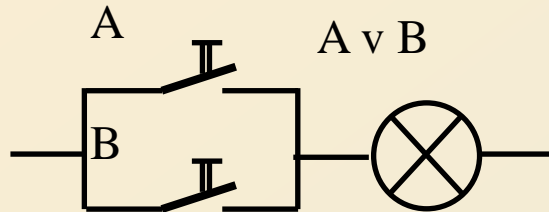


A	B	$A \wedge B$
0	0	0
0	1	0
1	0	0
1	1	1

A művelet elvégzése után az eredmény akkor igaz, ha az A és B is igaz volt, minden más esetben hamis.

Logikai műveletek

c) VAGY kapcsolat (OR)



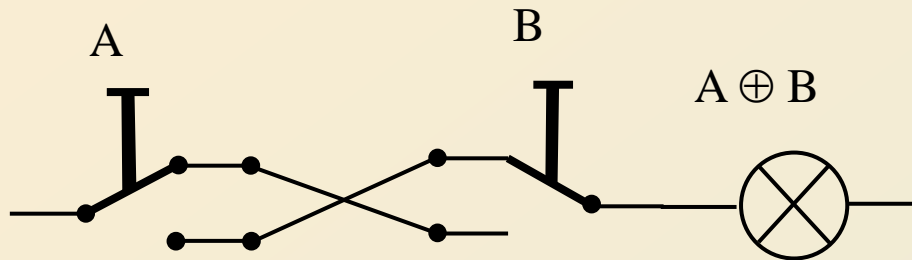
A	B	$A \vee B$
0	0	0
0	1	1
1	0	1
1	1	1

Az eredmény akkor igaz, ha vagy az A vagy a B is igaz volt, beleértve azt is, amikor mindkettő igaz.

Logikai műveletek

d) KIZÁRÓ VAGY kapcsolat (XOR)

Az eredmény kizárólag akkor igaz, ha vagy A vagy B igaz volt. Ez a kapcsolat kizárja azt az esetet, amikor mindkettő igaz.

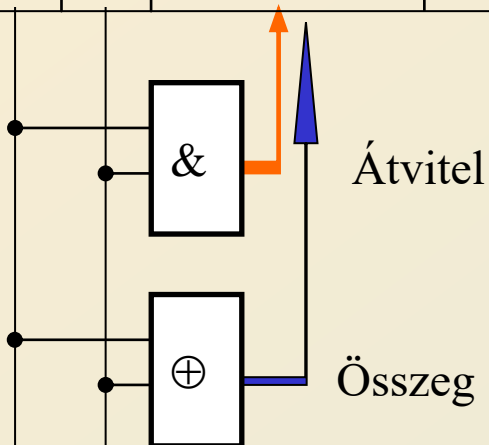


A	B	$A \oplus B$
0	0	0
0	1	1
1	0	1
1	1	0

Aritmetikai műveletek

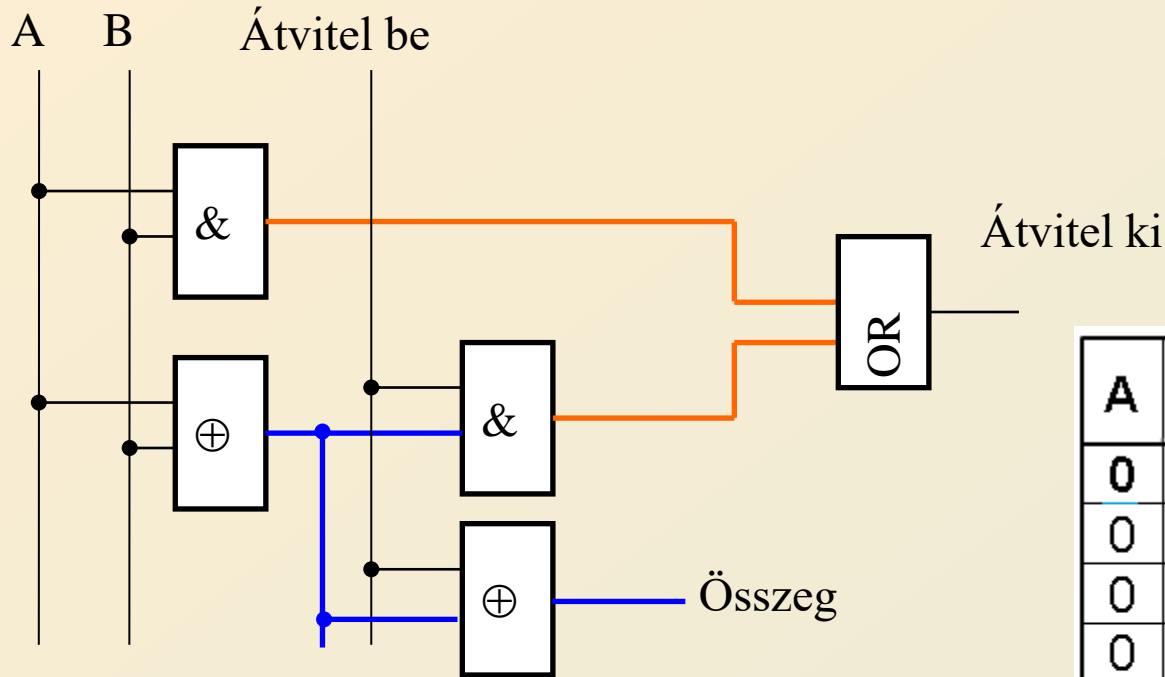
a) Összeadás (1 bites)

A	B	A + B	A & B	A \oplus B
0	0	00	0	0
0	1	01	0	1
1	0	01	0	1
1	1	10	1	0



Aritmetikai műveletek

b) Összeadás (Teljes összeadó)



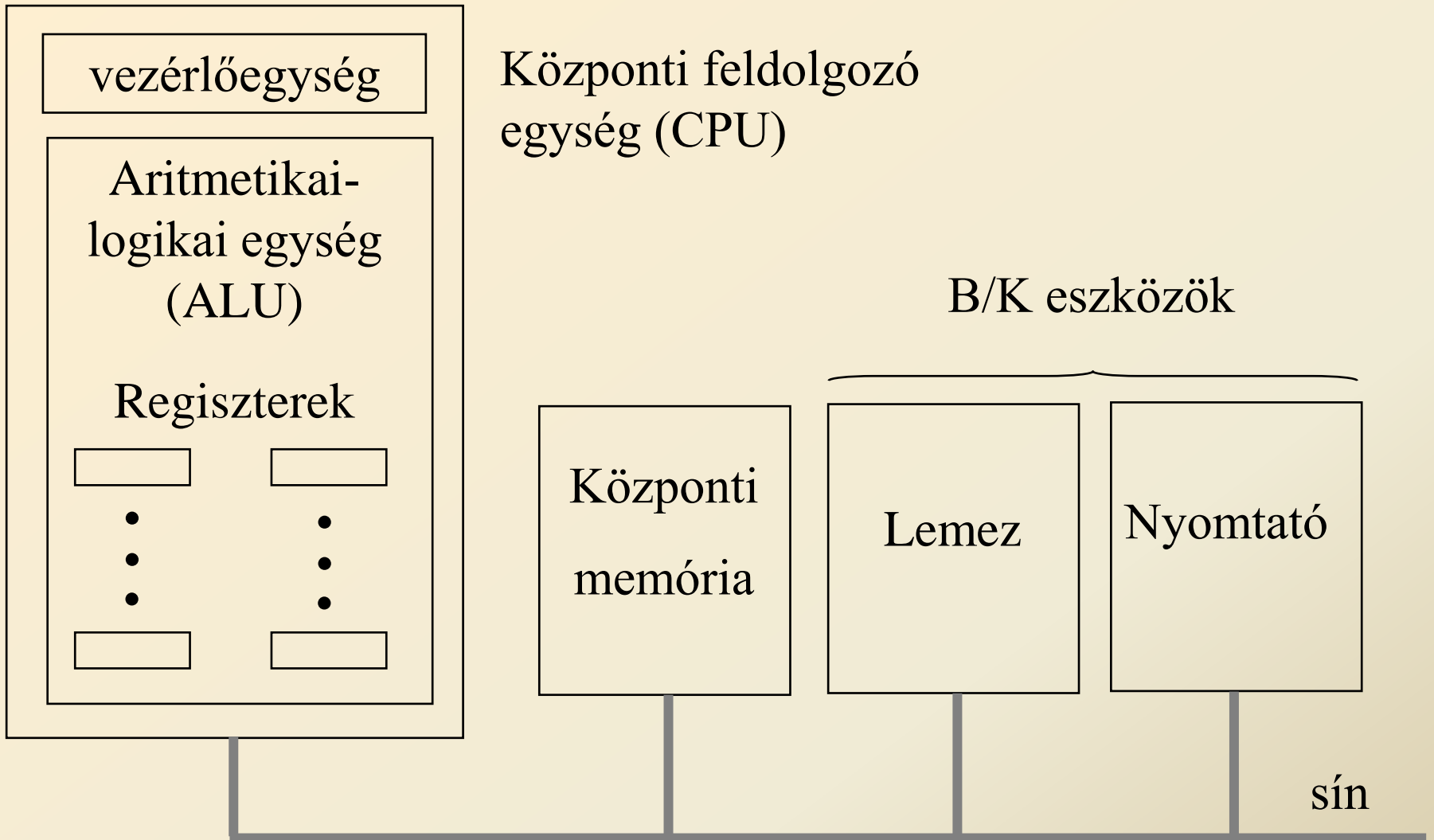
A	B	Átvitel be	Összeg	Átvitel ki
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Adattípusok

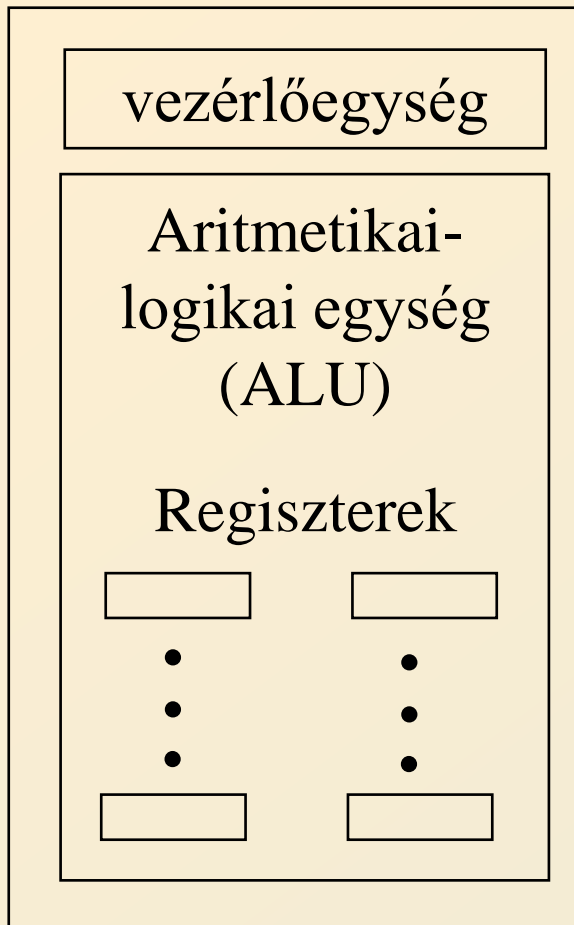
Alapkérdés: mit támogat a hardver (milyen utasítások vannak)? Ami nincs (pl. dupla pontosságú egész aritmetika), azt szoftveresen kell megcsinálni.

Numerikus típusok:

- előjel nélküli és előjeles egész számok (**8, 16, 32, 64 bites**).
- lebegőpontos számok (**32, 64, néha 128 bites**),
- binárisan kódolt decimális számok: decimális aritmetika

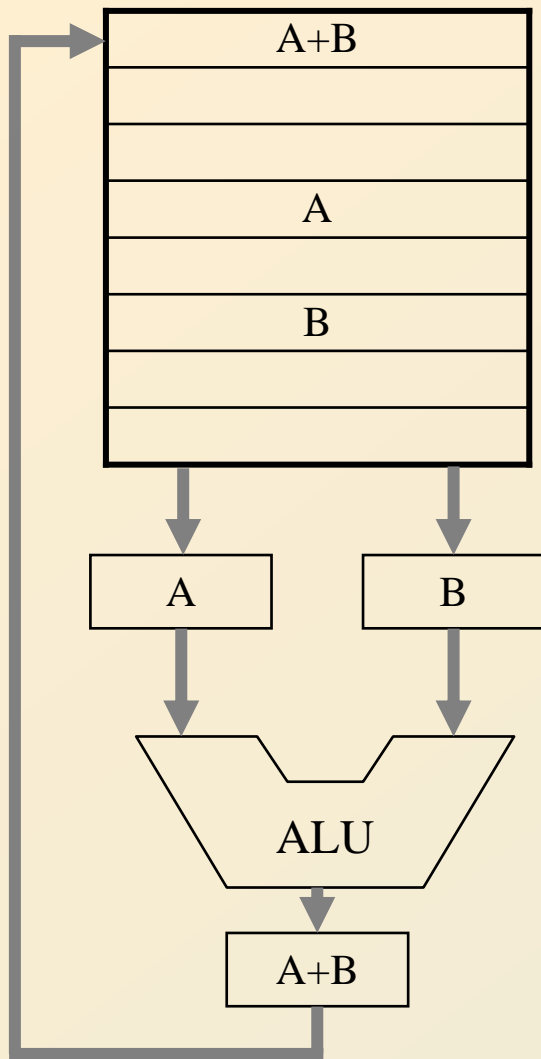


Egyszerű sín alapú számítógép



CPU feladata: a memóriában tárolt program végrehajtása. Részei:

- vezérlőegység, feladata: a program
 - utasításainak beolvasása,
 - az **ALU**, a regiszterek vezérlése,
- aritmetikai-logikai egység (**ALU**), feladata: az utasítások végrehajtása,
- regiszter készlet, feladata: részeredmények, vezérlő információk tárolása. A legfontosabbak:
 - utasításszámláló
(Program Counter): **PC**,
 - utasításregiszter
(Instruction Register): **IR**,
- adatút (data path).



Adatút (data path).

- A regiszter készletből feltöltődik az **ALU** két bemenő regisztere
- **ALU**
- Az eredmény az **ALU** kimenő regiszterébe kerül
- Az **ALU** kimenő regiszteréből a kijelölt regiszterbe kerül az eredmény

Nem biztos, hogy az **ALU** be- és kimenő regiszterei tényleges regiszterként vannak kialakítva.